ThunderNet: Towards Real-time Generic Object Detection

Zheng Qin, Zeming Li et.al

wyuzyf,April 2019

1 Abstract

提出实时通用的 two-stage detector，以及一个轻量的 two-stage detector ： ThunderNet。设计了两个有效的结构：Context Enhancement Module and Spatial Attention Module。在 ARM 移动端可以达到 ***24.1fps***，作者说这是第一个实时的检测框架。

2 Introduction

基于 CNN 的 detectors 可以被拆分成两个部分：**backbone part which extracts features for the image and the detection part which detects object instances in the image**。Object detection 需要大的 receptive filed 和低级特征得到精细的定位。本文的目标是在不损害精度的情况下继续压缩网络。

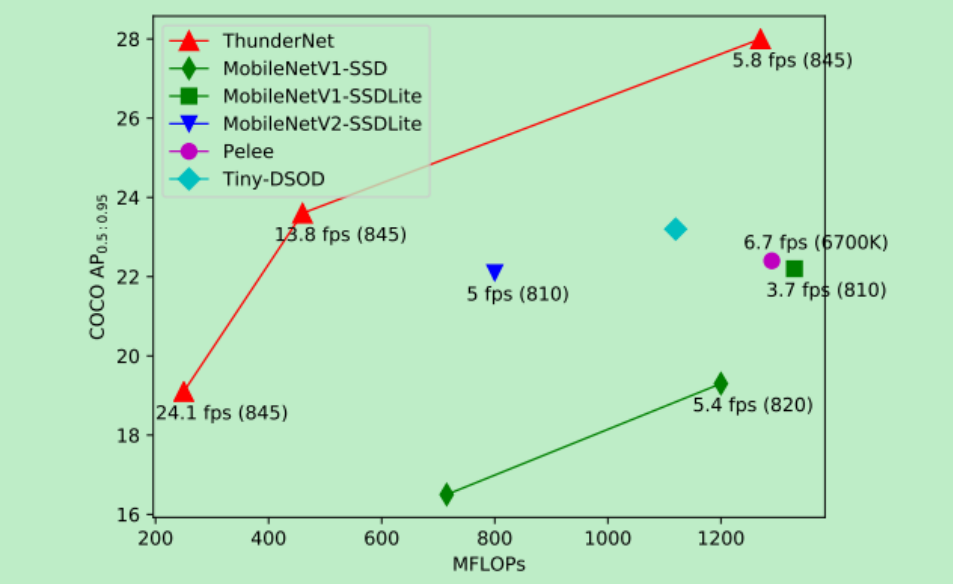

Figure 1. Comparison of ThunderNet and previous lightweight detectors on COCO test-dev[1]. ThunderNet achieves improvements in both accuracy and efficiency.

图 1: 各轻量网络的比较

two-statge：Light Head R-CNN 可以在 GPU 达到实时的检测。However, when coupled with a small backbone, Light-Head R-CNN still spends more computation on the detection part than the backbone, which leads to a mismatch between a weak backbone and a strong detection part. 这种不平衡会导致冗余，以及更容易过拟合。

one-stage：one-stage detectors directly predict bounding boxes and class probabilities。However,

as one-stage detectors do not conduct RoI-wise feature extraction and recognition, their results are coarser than two-stage detectors。也有 one-stage 的网络压缩，如 Tiny-dsod，pelee 等。

It inspires us to rethink: can two-stage detectors surpass one-stage detectors in real-time detection?

In this paper, we propose a lightweight two-stage generic object detector named ThunderNet.。在 backbone 阶段，设计了一个轻量网络 SNet；在 detection 阶段，遵循 Light-Head R-CNN, and further compress RPN and R-CNN subnet。为了防止 small backbone 导致性能下降，设计了两个 blocks，Context Enhancement Module (CEM) and Spatial Attention Module (SAM)。CEM 组合多个 scales 的 local and global 的 feature maps，SAM 用 RPN 学到的信息微调 feature 的分布。

ThunderNet surpasses prior lightweight one-stage detectors with significantly less computational cost on PASCAL VOC [5] and COCO [18] benchmarks. ThunderNet outperforms Tiny-DSOD [13] with only 42% of the computational cost and obtains gains of 6.5 mAP on VOC and 4.8 AP on COCO under similar complexity. Without bells and whistles, ThunderNet runs in *real time* on ARM (24.1 fps) and x86 (47.3 fps) with MobileNet-SSD level accuracy. To the best of our knowledge, this is the *first* real-time detector and the *fastest* single-thread speed reported on ARM platforms. *These results have demonstrated the effectiveness of two-stage detectors in real-time object detection.*

图 2: 效果

（上面这段话写的如此的霸气!!!!!）

3 Related Wok

其中的一个亮点是，可以为 object detector 设计专用的 backbone 网络，如 DetNet，Tiny-dsod，Pelee 等。

4 ThunderNet

- **Backbone Part**

Input Resolution：two-stage detectors 的输入分辨率一般是比较大的，为了提高 inference 的速度，ThunderNet 统一输入分辨率：320*320 pixels。在后续的实践中，发现输入分辨率应该和 backbone 的大小匹配，后面会有 Ablation Experiments。

Backbone Network：通常会使用在 ImageNet 预训练好的分类模型改进后，用于检测框架，但这简单的转换不是最佳的。

Receptive filed：大的感受野在目标检测任务是非常有用的。

Early-stage and late-stage features：In the backbone, early-stage feature maps are larger with low-level features which describe spatial details, while late-stage feature maps are smaller with high-level

features which are more discriminative。在 low-level 的特征中，目标的 localization 信息是比较精确的，所以 early-stage 是非常重要的。**we start from ShuffleNetV2, and build a lightweight backbone named *SNet* for real-time detection.** 。SNet49 for faster inference, SNet535 for better accuracy, and SNet146 for a better speed/accuracy trade-off。如下图，

Stage	Output Size	Layer		
		SNet49	SNet146	SNet535
Input	224×224	image		
Conv1	112×112	3×3, 24, s2	3×3, 24, s2	3×3, 48, s2
Pool	56×56	3×3 maxpool, s2		
Stage2	28×28 28×28	[60, s2] [60, s1]×3	[132, s2] [132, s1]×3	[248, s2] [248, s1]×3
Stage3	14×14 14×14	[120, s2] [120, s1]×7	[264, s2] [264, s1]×7	[496, s2] [496, s1] ×7
Stage4	7×7 7×7	[240, s2] [240, s1]×3	[528, s2] [528, s1]×3	[992, s2] [992, s1]×3
Conv5	7×7	1×1, 512	-	-
Pool	1×1	global avg pool		
FC		1000-d fc		
FLOPs		49M	146M	535M

Table 1. Architecture of the SNet backbone networks. SNet uses ShuffleNetV2 basic blocks but replaces all 3×3 depthwise convolutions with 5×5 depthwise convolutions.

图 3: backbone network

主要是增大了感受野和增加了 channels。

- **Detection Part**

Compressing RPN and Detection Head: 还是遵循 Light-Head R-CNN 的方法，为了获取更多的上下文信息，compress RPN by replacing the original 256-channel 3×3 convolution with a 5×5 depthwise convolution and a 256-channel 1×1 convolution. We increase the kernel size to enlarge the receptive field and encode more context information。增加 anchors 的 scales 和 aspect ratio，以及后续的减少计算量的操作。

Context Enhancement Module: Light-Head R-CNN applies Global Convolutional Network (GCN) to generate the thin feature map.CEM 的目标是合并多尺度的 local 和 global 上下文信息。

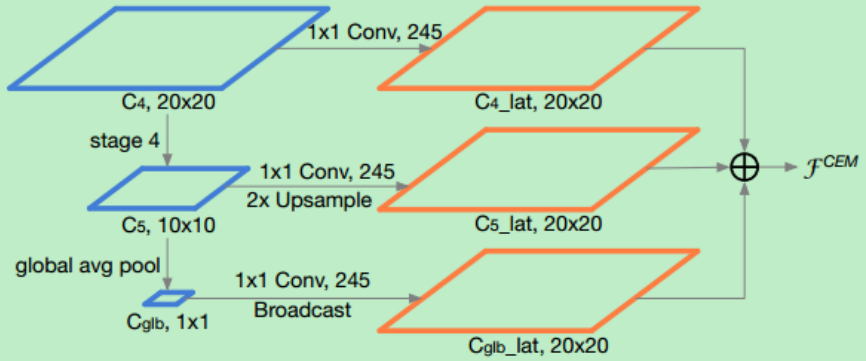

Figure 3. Structure of Context Enhancement Module (CEM). CEM combines feature maps from three scales and encodes more context information. It enlarges the receptive field and generates more discriminative features.

图 4: CEM

如上图所示，利用 global average pooling 减少参数，降低计算复杂度。然后用 1*1 卷积使得每个输出的 feature map 的维数相等。

Spatial Attention Module：在 ROI warping 期间，我们提取背景和前景的特征。相比较于大型的 backbone，thundernet 不容易找到合适的特征分布，所以作者设计了 SAM 模块。在 ROI 操作之前，手动的设计 feature map 的分布通过修改权重。RPN is trained to recognize foreground regions under the supervision of ground truths，所以前景更容易区分。下图所示为 SAM 公式和结构设计：

background features. SAM accepts two inputs: the intermediate feature map from RPN $\mathcal{F}^{RPN}$ and the thin feature map from CEM $\mathcal{F}^{CEM}$. The output of SAM $\mathcal{F}^{SAM}$ is defined as:

$$\mathcal{F}^{SAM} = \mathcal{F}^{CEM} \cdot sigmoid(\theta(\mathcal{F}^{RPN})). \quad (1)$$

图 5: SAM 公式

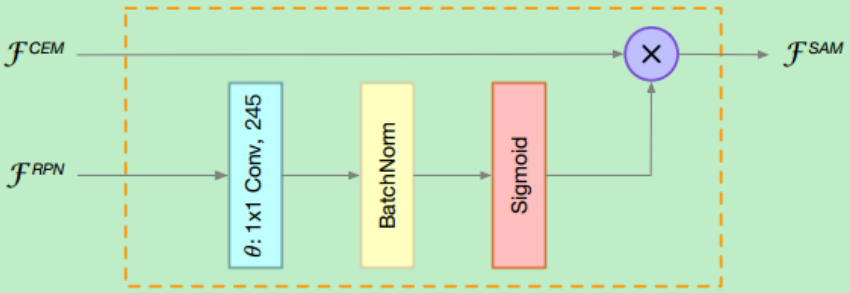

Figure 4. Structure of Spatial Attention Module (SAM). SAM leverages the information learned in RPN to refine the feature distribution of the feature map from Context Enhancement Module. The feature map is then used for RoI warping.

图 6: SAM 结构

5 Experiments

Model	Backbone	Input	MFLOPs	mAP
YOLOv2 [25]	Darknet-19	416 × 416	17400	76.8
SSD300* [19]	VGG-16	300 × 300	31750	77.5
SSD321 [6]	ResNet-101	321 × 321	15400	77.1
DSSD321 [6]	ResNet-101 + FPN	321 × 321	21200	78.6
R-FCN [4]	ResNet-50	600 × 1000	58900	77.4
Tiny-YOLO [25]	Tiny Darknet	416 × 416	3490	57.1
D-YOLO [21]	Tiny Darknet	416 × 416	2090	67.6
MobileNet-SSD [31]	MobileNet	300 × 300	1150	68.0
Pelee [31]	PeleeNet	304 × 304	1210	70.9
Tiny-DSOD [13]	DDB-Net + D-FPN	300 × 300	1060	72.1
ThunderNet (*ours*)	SNet49	320 × 320	**250**	70.1
ThunderNet (*ours*)	SNet146	320 × 320	**461**	**75.1**
ThunderNet (*ours*)	SNet535	320 × 320	1287	**78.6**

Table 2. Evaluation results on VOC 2007 test. ThunderNet surpasses competing models with significantly less computational cost.

Model	Backbone	Input	MFLOPs	AP	AP_{50}	AP_{75}
YOLOv2 [25]	Darknet-19	416 × 416	17500	21.6	44.0	19.2
SSD300* [19]	VGG-16	300 × 300	35200	25.1	43.1	25.8
SSD321 [6]	ResNet-101	321 × 321	16700	28.0	45.4	29.3
DSSD321 [6]	ResNet-101 + FPN	321 × 321	22300	28.0	46.1	29.2
Light-Head R-CNN [20]	ShuffleNetV2*	800 × 1200	5650	23.7	-	-
MobileNet-SSD [11]	MobileNet	300 × 300	1200	19.3	-	-
MobileNet-SSDLite [28]	MobileNet	320 × 320	1300	22.2	-	-
MobileNetV2-SSDLite [28]	MobileNetV2	320 × 320	800	22.1	-	-
Pelee [31]	PeleeNet	304 × 304	1290	22.4	38.3	22.9
Tiny-DSOD [13]	DDB-Net + D-FPN	300 × 300	1120	23.2	40.4	22.8
ThunderNet (*ours*)	SNet49	320 × 320	**262**	19.1	33.7	19.6
ThunderNet (*ours*)	SNet146	320 × 320	**473**	**23.6**	40.2	**24.5**
ThunderNet (*ours*)	SNet535	320 × 320	1300	**28.0**	**46.2**	**29.5**

Table 3. Evaluation results on COCO test-dev. ThunderNet with SNet49 achieves MobileNet-SSD level accuracy with 22% of the FLOPs. ThunderNet with SNet146 achieves superior accuracy to prior lightweight one-stage detectors with merely 40% of the FLOPs. ThunderNet with SNet535 rivals large detectors with significantly less computational cost.

图 7: Results

如上图所示，ThunderNet 在各个数据集的表现都是非常亮眼的，达到了 SOAT。

Model	ARM	CPU	GPU
Thunder w/ SNet49	24.1	47.3	267
Thunder w/ SNet146	13.8	32.3	248
Thunder w/ SNet535	5.8	15.3	214

Table 9. Inference speed in fps on Snapdragon 845 (ARM), Xeon E5-2682v4 (CPU) and GeForce 1080Ti (GPU).

图 8: Inference

如上图所示，推理速度是很快的。